Roll no:- DS5B-2101

Q1:- Considering left as dependent variable in HR dataset, split the dataset according to your last digit of roll no. Here my roll no is ending with 1, the ratio will be 71, 29. The accuracy of the model is as follows:-

Here we are using Logistic Regression By Pyspark

In [1]:
```
!pip install pyspark
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/publ
ic/simple/
Collecting pyspark
  Downloading pyspark-3.2.1.tar.gz (281.4 MB)
     |████████████████████████████████| 281.4 MB 40 kB/s
Collecting py4j==0.10.9.3
  Downloading py4j-0.10.9.3-py2.py3-none-any.whl (198 kB)
     |████████████████████████████████| 198 kB 75.0 MB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.2.1-py2.py3-none-any.whl size=281853642
sha256=2f44a11c4a72385dab59265a0f6253b9380c9807b4b3e6e1331a3e8c8c7e329c
  Stored in directory: /root/.cache/pip/wheels/9f/f5/07/7cd8017084dce4e93e84e92efd1e1d53
34db05f2e83bcef74f
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.3 pyspark-3.2.1
```

Creat Spark session

Roll no:- DS5B-2101

In [26]:
```
from pyspark.sql import SparkSession
session = SparkSession.builder.appName("HR_Dataset").getOrCreate()
data = session.read.csv("HR comma.csv", header = True, inferSchema = True)
```

In [27]:
```
data.show(10)
```

```
+------------------+---------------+--------------+--------------------+---------------
--+--------------+----+--------------------+-----+------+
|satisfaction_level|last_evaluation|number_project|average_montly_hours|time_spend_compa
ny|Work_accident|left|promotion_last_5years|sales|salary|
+------------------+---------------+--------------+--------------------+---------------
--+--------------+----+--------------------+-----+------+
|              0.38|           0.53|             2|                 157|
3|             0|   1|                   0|sales|   low|
|               0.8|           0.86|             5|                 262|
6|             0|   1|                   0|sales|medium|
|              0.11|           0.88|             7|                 272|
4|             0|   1|                   0|sales|medium|
|              0.72|           0.87|             5|                 223|
5|             0|   1|                   0|sales|   low|
|              0.37|           0.52|             2|                 159|
3|             0|   1|                   0|sales|   low|
|              0.41|            0.5|             2|                 153|
3|             0|   1|                   0|sales|   low|
|               0.1|           0.77|             6|                 247|
4|             0|   1|                   0|sales|   low|
|              0.92|           0.85|             5|                 259|
5|             0|   1|                   0|sales|   low|
```

```
|              0.89|             1.0|             5|                224|
5|            0|   1|                  0|sales|   low|
|              0.42|            0.53|             2|                142|
3|            0|   1|                  0|sales|   low|
+------------------+----------------+--------------+--------------------+---------------
--+-------------+----+--------------------+-----+------+
only showing top 10 rows
```

In [28]:
```
data.columns
```

Out[28]:
```
['satisfaction_level',
 'last_evaluation',
 'number_project',
 'average_montly_hours',
 'time_spend_company',
 'Work_accident',
 'left',
 'promotion_last_5years',
 'sales',
 'salary']
```

In [29]:
```
from pyspark.ml.feature import VectorAssembler, StringIndexer, OneHotEncoder
str_idx = StringIndexer(inputCols = ['sales','salary'], outputCols = ["newsales", "newsa
```

In [30]:
```
one_hot = OneHotEncoder(inputCols = ["newsales","newsalary"], outputCols = ["newsales_on
```

In [31]:
```
vec_ass = VectorAssembler(inputCols = ['satisfaction_level','last_evaluation','number_pr
```

Now by Logistic Regression

Roll no:- DS5B-2101

In [32]:
```
from pyspark.ml.classification import LogisticRegression
lg_rg = LogisticRegression(featuresCol= "all_features", labelCol = "left")
```

Adding Pipelline

Roll no:- DS5B-2101

In [33]:
```
from pyspark.ml import Pipeline
mypipeline = Pipeline(stages = [str_idx, one_hot, vec_ass, lg_rg])
```

Data split into 71 & 29

In [34]:
```
training, test = data.randomSplit([0.71, 0.29])
```

In [35]:
```
lg_rg_model = mypipeline.fit(training)
```

In [36]:
```
result = lg_rg_model.transform(test)
```

In [37]:
```
result.show(4, truncate = False)
```

```
+------------------+----------------+--------------+--------------------+---------------
--+-------------+----+--------------------+-----------+------+--------+---------+------
---------+----------------+-----------------------------------------------------------+----
-----------------------------------+-----------------------------------------+----------
+
|satisfaction_level|last_evaluation|number_project|average_montly_hours|time_spend_compa
ny|Work_accident|left|promotion_last_5years|sales      |salary|newsales|newsalary|newsal
```

```
es_onehot|newsalary_onehot|all_features                                                       |rawP
rediction                                    |probability                                   |prediction
|
+-----------------+---------------+-------------+-------------------+---------------
--+-------------+----+--------------------+-----------+------+--------+---------+------
---------+----------------+---------------------------------------------------------+----
------------------------------------+----------------------------------------+----------
+
|0.09              |0.62            |6             |294                 |4
  |0              |1   |0                   |accounting |low   |7.0     |0.0       |(9,
[7],[1.0])  |(2,[0],[1.0])   |(18,[0,1,2,3,4,14,16],[0.09,0.62,6.0,294.0,4.0,1.0,1.0])|
[-1.0464291635344414,1.0464291635344414]|[0.25991138875030884,0.7400886112496912]|1.0
    |
|0.09              |0.62            |6             |294                 |4
  |0              |1   |0                   |accounting |low   |7.0     |0.0       |(9,
[7],[1.0])  |(2,[0],[1.0])   |(18,[0,1,2,3,4,14,16],[0.09,0.62,6.0,294.0,4.0,1.0,1.0])|
[-1.0464291635344414,1.0464291635344414]|[0.25991138875030884,0.7400886112496912]|1.0
    |
|0.09              |0.77            |5             |275                 |4
  |0              |1   |0                   |product_mng|medium|5.0     |1.0       |(9,
[5],[1.0])  |(2,[1],[1.0])   |(18,[0,1,2,3,4,12,17],[0.09,0.77,5.0,275.0,4.0,1.0,1.0])|
[-0.6795278770005595,0.6795278770005595]|[0.3363666837019514,0.6636333162980486] |1.0
    |
|0.09              |0.77            |5             |275                 |4
  |0              |1   |0                   |product_mng|medium|5.0     |1.0       |(9,
[5],[1.0])  |(2,[1],[1.0])   |(18,[0,1,2,3,4,12,17],[0.09,0.77,5.0,275.0,4.0,1.0,1.0])|
[-0.6795278770005595,0.6795278770005595]|[0.3363666837019514,0.6636333162980486] |1.0
    |
+-----------------+---------------+-------------+-------------------+---------------
--+-------------+----+--------------------+-----------+------+--------+---------+------
---------+----------------+---------------------------------------------------------+----
------------------------------------+----------------------------------------+----------
+
only showing top 4 rows
```

Evaluation of Model

Roll no:- DS5B-2101

In [38]:

```python
evaluation = ["f1","accuracy","weightedPrecision","weightedRecall", "weightedTruePositiv
for i in evaluation:
  from pyspark.ml.evaluation import MulticlassClassificationEvaluator
  eval = MulticlassClassificationEvaluator(predictionCol="prediction", labelCol= "left",
  print(i, ":", eval.evaluate(result))
```

```
f1 : 0.7670945018121722
accuracy : 0.787943769359066
weightedPrecision : 0.765798263974633
weightedRecall : 0.7879437693590661
weightedTruePositiveRate : 0.7879437693590661
weightedFalsePositiveRate : 0.5129530187585594
weightedFMeasure : 0.7670945018121722
truePositiveRateByLabel : 0.9246404002501564
falsePositiveRateByLabel : 0.6496496496496497
precisionByLabel : 0.8200221852468109
recallByLabel : 0.9246404002501564
fMeasureByLabel : 0.8691945914168137
logLoss : 0.42502890882986544
hammingLoss : 0.212056230640934
```